durée 2h00, aucun document autorisé

notation sur 24 points ramenés sur 21 points ensuite

Exercice 1 chiffrement à clefs publiques (sur 4 points)

On appelle E un procédé de chiffrement à clef publique et D le procédé de déchiffrement associé. On suppose qu'il existe un procédé de signature associé à E que l 'on notera S. On notera VS le procédé de vérification de signature associé.

On suppose que toutes les personnes intervenant dans cet exercice ont chacune un couple (clef privée, clef publique) correspondant aux procédés cités ci-dessus. Par souci de simplification, on supposera que le même couple peut servir indifféremment aux opérations de chiffrement ou de signature.

Répondez aux questions suivantes directement sur l'énoncé :

correction: cf cours. A noter qu'il faut indiquer à qui appartient la clef en jeu dans chaque étape. « La clef publique » est une réponse fausse. « La clef publique de Bob » est une réponse juste à la question 1.

Question 1: Alice veut envoyer un message chiffré à Bob, avec quelle clef doit-elle le chiffrer ?

A l'arrivée, quelle clef, Bob doit-il utiliser pour déchiffrer le message ?

Question 2: Alice veut envoyer un message signé à Bob, avec quelle clef doit-elle le signer ?

A l'arrivée, quelle clef, Bob doit-il utiliser pour vérifier la signature du message ?

Question 3: Alice veut envoyer un message chiffré et signé à Bob, avec quelle clef doit-elle

le chiffrer ?

Le signer ?

A l'arrivée, quelle clef, Bob doit-il utiliser pour déchiffrer le message ?

Pour vérifier la signature ?

Question 4: Alice veut envoyer un message chiffré et signé à Bob, Gérard, Jackie, Ahmed, ... (25 destinataires) avec quelle clef doit-elle le chiffrer ? Le signer ?

Exercice 2 Diffie-Hellman (6 points = 2+2+2)

question 1

Quel est le but de l'algorithme de Diffie-Hellman ?

Cet algorithme permet à deux personnes de se mettre d'accord sur un secret commun en échangeant des messages publiques qu'un espion peut voir.

Remarque sur une réponse fausse que l'on a trouvé dans plusieurs copies : le secret commun pourra avoir de nombreux usages. Il pourra servir notamment à chiffrer les communications entre les deux personnes. Le but de l'algorithme reste néanmoins l'accord sur un secret commun.

question 2

Les données suivantes sont publiques :

- p un grand nombre premier,
- G un groupe multiplicatif de cardinal p-1
- g un générateur de G

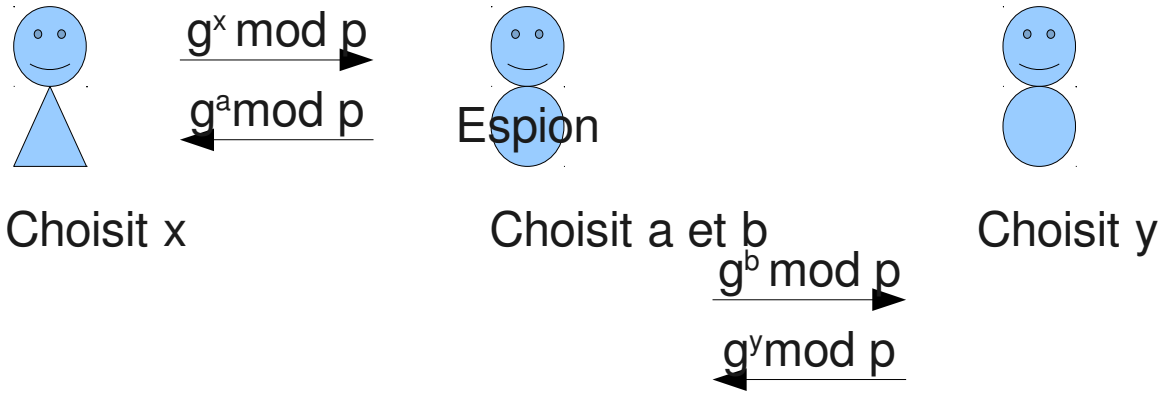
Décrivez l'algorithme de Diffie Hellman (description détaillée de chacune des étapes).

Cf cours. Ne pas oublier le modulo p dans les calculs (« g^x mod p » et non pas « g^x »).

question 3

Alice et Bob veulent communiquer de façon sûre à travers un réseau non sûr. Ils décident d'utiliser Diffie-Hellman. L'espion possède un contrôle total du réseau : il peut lire et modifier tout ce qui est y passe. Expliquez quelles sont les faiblesses de Diffie-Hellman dans ce contexte.

Eléments de correction: si l'espion peut tout lire et modifier, il peut réaliser une attaque man in the middle en se faisant passer pour Alice auprès de Bob et pour Bob auprès d'Alice. Il peut alors espionner et modifier tout ce qu'Alice et Bob échangent :



A noter : affirmer simplement « il peut réaliser une attaque ManInTheMiddle » sans la justifier par l'algorithme de l'attaque (schéma ci-dessus) n'apportait qu'une faible partie des points. Rappelez vous : en crypto, il faut argumenter ce que l'on affirme.

Exercice 3 hachage cryptographique (4 =2+2 points)

question 1

Quelles sont les propriétés que doivent vérifier les fonctions de hachage cryptographiques pour pouvoir être utilisées dans le cadre d'applications cryptographiques ?

Une fonction de hachage cryptographique est une fonction h prenant des arguments de taille quelconque et fournissant un résultat de taille fixe. H doit vérifier les propriétés suivantes (traduire « difficile » par « matériellement impossible avec les moyens du moment) :

- ***h est à sens unique : connaissant h(m), on ne doit pas pouvoir pratiquement retrouver m***
- ***elle doit résister aux attaques en seconde préimages: connaissant h(m), il doit être difficile de trouver m' tel que h(m')=h(m)***

- ***collisions difficiles: il doit être difficile de trouver m et m' tels que h(m)=h(m'). A noter que le fait que l'ensemble d'arrivée soit plus petit que l'ensemble de départ fait que h n'est pas injective. Il existe forcément des valeurs m et m' tels que h(m)=h(m')***
- ***dispersion: changer un bit du message m change beaucoup de bits du résultat***

Donnez deux exemples d'applications utilisant les fonctions de hachage à sens unique dans lesquelles il est important que ces propriétés soient vérifiées. Pour chaque application, vous expliquerez en quoi ces propriétés interviennent

gestion des mots de passe : on stocke e=h(m) et quand un utilisateur veut s'authentifier avec un mot de passe m', on calcule h(m') et on le compare à e. La solidité de la méthode repose sur le fait que h est à sens unique et résistante aux attaques en seconde préimage.

Intégrité des fichiers : on transmet par un moyen sur e=h(m) et on transfère par un moyen rapide m. à l'arrivée, on compare e avec h(m reçu) pour vérifier si le m reçu n'a pas été modifié par une personne malveillante. La solidité de la méthode reponse sur la résistance aux attaques en seconde préimage.

signature de gros fichiers : on signe l'emprunte du fichier plutôt que le fichier lui-même. Même problématique que ci-dessus.

question 2

Exercice 4 hachage cryptologique (7 points = 2+2+3)

On considère l'application suivante du hachage cryptographique. On suppose que h est une fonction de hachage cryptographique que tout le monde (Alice, le serveur, l'espion, …) connaissent.

- Alice choisit un nombre g
- elle calcule h(g), h(h(g)), …, h(h(h(h(h(h(h(h(h(h(g))))))))))=h^{11}(g)
- elle transmet[1] h^{11}(g) au serveur qui le stocke tel quel.

L'espion, de son côté, peut espionner tout ce qui passe sur le réseau. Il souhaite se connecter à distance au serveur en se faisant passer pour Alice.

question 1

Pour s'authentifier à distance, Alice transmet h^{10}(g) au serveur.

- Est-ce un authentification solide ?

C'est une authentification solide car on ne peut déduire h^{10}(g) à partir de h^{11}(g) (comme toujours, l'affirmation « c'est une authentification solide » non justifiée ne rapportait pas ou peu de points). Alice est donc seule à pouvoir fournir h^{10}(g).

Pour s'authentifier à distance une deuxième fois, Alice transmet h^{10}(g) au serveur.

- Est-ce un authentification solide ?

1 Supposons qu'elle tape directement cette valeur sur le clavier du serveur

Ce n'est pas une authentification solide car l'espion a déjà vu passer $h^{10}(g)$.

question 2

Proposez un algorithme d'authentification sûr qui s'appuie sur les résultats de la question 1 et qui résiste à un espion qui peut lire[2] tout ce qui passe sur le réseau. La seule fonction cryptographique que votre algorithme est autorisé à utiliser est le fonction h.

le problème mis en évidence à la question précédente est lié à la réutilisation de $h^{10}(g)$. Si on ne l'avait utilisé qu'une fois, on aurait eu une authentification solide. Pb: comment s'authentifier la deuxième fois ? Réponse en fournissant $h^9(g)$. C'est le mécanisme des mots de passe jetable « OTP (One Time Password, RFC2289, sur une idée de Leslie Lamport) :

- ***on choisit g***
- ***on fournit $h^{11}(g)$ au serveur***
- ***première authentification :***
  - ***Alice fournit $h^{10}(g) = m$ au serveur***
  - ***le serveur valide l'authentification en vérifiant que $h(m)=h^{11}(g)$***
  - ***le serveur mémorise $h^{10}(g)$***
- ***seconde authentification:***
  - ***Alice fournit $h^9(g) = m$ au serveur***
  - ***le serveur valide l'authentification en vérifiant que $h(m)=h^{10}(g)$***
  - ***le serveur mémorise $h^9(g)$***
- **…**

Évidemment, une fois qu'on a fourni g, il faut choisir un nouveau nombre g et transmettre $h^{11}(g)$ au serveur de façon sûre. On peut bien sur choisir des valeurs plus grandes que 11 pour éviter d'avoir à refaire ça trop souvent.

Exercice 5 hachage cryptologique (5 points = 1+1+1+2)

Dans cet exercice nous étudions des fonctions de hachage à sens unique. Pour simplifier la présentation, ainsi que les essais qui sont demandés, nous allons les décrire comme retournant une empreinte sur 8 bits. Il est clair que cela est insuffisant pour des questions de sécurité mais toutes les descriptions qui suivent peuvent être étendues pour générer des empreintes de longueur quelconques.

Si le message M à hacher n'a pas une longueur qui est un multiple de 8 bits, la fonction de hachage commence par le compléter avec un bit 1 suivi d'autant de bits 0 que nécessaire pour que le message ainsi complété contienne un multiple de 8 bits. Notons M' le message ainsi complété après cette étape. M' est alors découpé en blocs de 8 bits consécutifs. Notons B_1,B_2,...,B_q ces q blocs, $q \geqslant 1$. On peut alors décrire les deux fonctions de hachage étudiées dans cet exercice.

$$\mathbf{h\_1(M) = B\_1 \oplus B\_2 \ ... \ B\_l}$$

2 Mais pas modifier

avec ⊕ désignant le XOR bit à bit (Rappel~: 0 ⊕ 0= 1 ⊕ 1 =0 et 0 ⊕ 1 = 1 ⊕ 0 = 1).

Pour décrire h_2, notons A = a_1a_2... a_8 = B_1 ⊕ B_2 ⊕ ... ⊕ B_l (où a_i ∈ {0,1 }). Avec ces notations, on a~: h_2(M) = c_1... c_8 avec c_i = a_i ⊕ a_{i+1}, pour i=1,..., 7 et c_8 = a_8 ⊕ 0.

- Répondez aux questions suivantes en justifiant clairement vos calculs. :
  - Soit M=001011100001, calculez h_1(M) et h_2(M).
  - Construisez un message M tel que h_1(M) = 01011001.

prendre M=h_1(M) convient. On peut aussi choisir B1 quelconque et trouver ensuite B2 tel quel que B1 XOR B2=h_1(M) : B2=B1+h_1(M), … Il y a donc de nombreus solutions faciles à calculer.

  - Construisez un message M tel que h_2(M)=11001011

la formule c_8 = a_8 ⊕ 0. permet de trouver a_8

la formule c_7 = a_7 ⊕ a_8. permet de trouver a_7

...

  - Comment l'espion peut il attaquer h_1 et h_2 ? Que peut il faire ?

Dans votre réponse à la question précédente vous avez peut être utilisé le fait que les empreintes calculées ne sont que sur 8 bits. Qu'en est il de la sécurité de h_1 et h_2 si on les généralise naturellement pour produire des empreintes de 160 bits?

On a réussi à trouver facilement M à partir de h_1(M) et à partir de h_2(M). Les méthodes trouvées sont tellement simple qu'un être humain est capable de le faire en quelques minutes même pour 160 bits. Ces fonctions de hachage ne sont absolument pas sure. Il ne faut pas les utiliser.

Exercice 6 PKI (4 points = 2+2)

A quel problème répond une infrastructure de gestion de clefs (PKI) ?

Les algorithmes à clefs publiques sont sûrs. Le problème est d'avoir la preuve qu'une clef publique est bien la clef publique de la personne avec laquelle on veut communiquer. Une PKI résout ce problème en mettant en place une autorité de confiance qui certifiera les clefs publiques en précisant pour chaque clef de qui elle est la clef et si elle est valide.

Décrivez sa structure.

Une PKI comprend trois éléments obligatoires :

- ***une autorité d'enregistrement chargée de vérifier l'identité du possesseur de la clef publique en appliquant la politique définie par l'autorité de certification. L'utilisateur qui veut obtenir un certificat s'adresse à elle***
- ***une autorité de certification qui est une autorité de confiance reconnue par une communauté d'utilisateurs. Elle délivre et gère les certificats (« clefs publiques + identités » signées), maintient une liste des certificats révoqués.***

- ***Un service de publication qui met à la disposition de la communauté les certificats et liste aussi ceux qui sont révoqués.***